LLNL-TR-458089

LAWRENCE
LIVERMORE
NATIONAL
LABORATORY

Uncertainty Quantification in the Fusion Simulation Project Verification and Validation Activity

J. A. Hittinger, B. I. Cohen, R. I. Klein

October 5, 2010

Disclaimer

This document was prepared as an account of work sponsored by an agency of the United States government. Neither the United States government nor Lawrence Livermore National Security, LLC, nor any of their employees makes any warranty, expressed or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States government or Lawrence Livermore National Security, LLC. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States government or Lawrence Livermore National Security, LLC, and shall not be used for advertising or product endorsement purposes.

This work performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344.

Uncertainty Quantification in the Fusion Simulation Project
Verification and Validation Activity

Jeffrey A. Hittinger, Bruce I. Cohen, and Richard I. Klein
Lawrence Livermore National Laboratory
Livermore, CA 94551

Abstract

Uncertainty quantification (UQ) is an important component of any investigation of the behavior of a complex, multi-physics system. UQ is the complete study of the accuracy, reliability, and errors in deriving scientific inferences. UQ impacts both the experimental and modeling (analytical theory and numerical simulation) approaches to studying the behavior of complex systems. UQ is essential in quantifying the nature of model and code verification and validation. Although UQ is an evolving and expanding research area, there is much experience and many mature tools already in existence. Thus, a new undertaking like the Fusion Simulation Project, with goals to develop a suite of predictive computer models for complex magnetic fusion plasma behavior and to validate this suite against experimental observations, will need to make thorough use of existing UQ methodologies and motivate the development of new techniques. This document presents an overview of UQ including some examples of successful practices and some recommendations for defining UQ practices in the Fusion Simulation Project.

CONTENTS

ABSTRACT

1. INTRODUCTION

2. UNCERTAINTY QUANTIFICATION BASICS

3. LLNL UQ EXPERIENCE AND EXAMPLES OF USE

4. SUMMARY AND RECOMMENDATIONS

REFERENCES AND LITERATURE ACQUIRED

1. INTRODUCTION

Uncertainty quantification (UQ) is intended to be the comprehensive study of the accuracy, reliability, and errors in making scientific inferences. UQ applies quite generally to both experiment and analysis of all kinds, including simulation. The roots of UQ for simulations can be found in the science of experimental design. In experimental design, one wishes to quantify the dependencies and correlations of experimental measurements on various measurable attributes, including attributes that the experimentalist can control and vary. Sensitivity and error analyses are part of quantitatively understanding the results of experiments. When the number of independent, controllable attributes becomes large, then the experimentalist is confronted with the significant challenge of how to sample the complete multi-dimensional parameter space efficiently so as to capture all of the correlations and cross-correlations. Brute-force random sampling of the parameter space scales geometrically with the number of parameters that are varied. Trying to undertake UQ in simulations shares many of the same challenges as UQ in experiments, and some of the challenges are exacerbated because there can be many more parameters in the simulation models and the approximations inherent in the physical model may be difficult to quantify. In complex, multi-physics plasma simulations, there can be a large number of parameters that can lead to a computationally prohibitive large set of simulations if random sampling methodologies are employed.

In this overview, we define UQ in the context of the Fusion Simulation Project. UQ enters through both verification and validation of the physics models and the computations, as well as in the analysis of experimental data independent of the simulation effort. Because there has been a great deal of effort in UQ for both experiment and simulation and a significant increase in the development of UQ methods for simulations in recent years, there is a large and expanding literature, including good review articles. We do not attempt to review the field of UQ, nor do we attempt to define best practices because UQ methodologies are application dependent and are evolving. Instead we provide an overview with pointers to some of the literature that we deem useful and describe some examples of use to furnish guidance on successful practices.

The context of the Fusion Simulation Project (FSP) involves the development of a suite of simulations that embody the various physics models needed to model tokamak plasmas in a comprehensive fashion. The physics models will be coupled to one another through a common framework in varying degrees. Some of the existing simulation codes already couple a suite of physics models fairly tightly, while other calculations are very narrowly focused on a single physics component. To assess whether the FSP is successful in improving our understanding and modeling of plasma behavior in tokamak plasmas accurately, we must define quantitative metrics. Making quantitative comparisons of physics models and simulations to experimental data, i.e., validation efforts, obligates one to understand the accuracy and the errors in the measurements and the computations in a systematic way and to assess the correlations and cross-correlations of the physical quantities and parameters. UQ provides the tools and methodologies for the general validation effort.

Verification and validation are necessarily a multi-step process. We seek to compare the inferences of a physics model to experimental measurements, but there is the intervening numerical implementation of the physics model whose contributions to accuracy and error must be understood in a systematic and rigorous verification effort. UQ contributes to the study of accuracy and of errors in the verification process; the quantitative study of how accurately the numerical solution of the physics model corresponds to the true mathematical solution of the model equations is part of UQ.

The quantitative comparison of the physics model to experiment is futile without having a quantitative understanding of the accuracy and errors in the computations that approximate the physics model as well as a good understanding of the errors embodied in the experimental measurements. One can validate the computer calculation with respect to experiment and be misled with respect to understanding the quantitative comparison of the underlying physics model(s) to experiment, unless one has a good understanding of the effects of the approximations separating the numerical model and physical models.

In Sec. 2 of this overview we describe the basics of UQ in the context of verification and validation in general terms. This discussion includes some of the types of error to be investigated, strategies for sampling parameter space and reducing dimensionality, and methodologies for error and sensitivity analysis. We present some discussion of and references to UQ experience and examples of use in Sec. 3. A brief summary and recommendations on process are given in Sec. 4. References and a list of publications, presentations, and reports acquired follow.

2. UNCERTAINTY QUANTIFICATION BASICS

2.1 Definitions

We begin with several definitions. ***Uncertainty Quantification*** is the quantification of all sources of uncertainty that may affect an inference. A UQ analysis will produce a quantification of the confidence in a result or in some instances, a bounding of the error associated with a result. Closely related to UQ is ***Sensitivity Analysis (SA)***, which is the quantification of the sensitivity of a result to specific input data, e.g., initial conditions and model parameters. Typically, sensitivity analysis is a first step in developing a formal uncertainty estimate.

In the context of the FSP, we specialize to scientific inference based on physical models. We are fundamentally concerned with the comparison of physical models to experimental data, that is, ***Validation***. Both UQ and SA apply equally well to the scientific results of both experiments and simulations. To validate simulation results with experimental data rigorously, UQ for both should be determined; both types of data require “error bars” to evaluate properly the degree of agreement. SA results can then be used to identify those data for which better knowledge would the most leverage in affecting the comparison to experiment. Of course, if disagreement between experimental and simulation results is beyond the quantified uncertainty, it implies a deficiency in the assumed physical model for the intended purpose or a deficiency in the experimental data that is not well quantified, or both. The distinction is frequently difficult to unravel. Note that validation occurs within a use context; thus, to speak of a “validated model” or “validated code” without this context is meaningless.

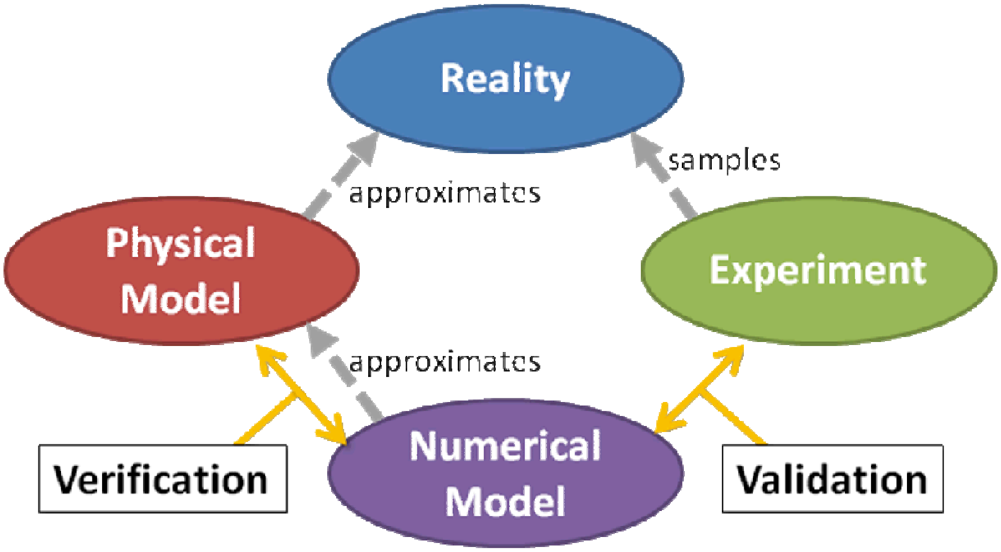
There are many different kinds of errors and uncertainty that affect the embodiment of the physics model in a computer code and that can potentially affect the comparisons of the results of the computations to experimental observations. These include, but are not limited to:

- systematic and stochastic measurement error (epistemic uncertainty);

- limitations of theoretical and phenomenological models, i.e., what phenomena are included and what are omitted. These limitations can introduce systematic errors that can be difficult to quantify;
- limitations of the representations of the data and models (how models and data are simplified for use in analysis);
- accuracy of the data used in physical models
- accuracy of computation and approximation, including both deterministic and stochastic approximation errors, iteration errors, and finite-precision arithmetic errors;
- random phenomena (aleatoric uncertainty).

Of these, one set is distinguished for UQ of numerical simulations: errors resulting from discrete approximation. As shown in Figure 1, a simulation code is the embodiment of a discrete, finite-precision approximation to a physical model. The physical model is typically a set of ordinary or partial differential equations that themselves are an approximation of physical reality. Ideally, we would like to compare experimental results directly with our physical models, but instead we compare with the surrogate numerical model. Thus, we must carefully include the uncertainty due to numerical approximation in an UQ estimate based on simulation data.

Figure 1. Relationships between experiments, models, verification, and validation.



Verification is the technical term used to describe the assessment of the accuracy of a numerical model, that is, to demonstrate that the code approximates the intended physical model. It has a broader meaning in software quality assurance to mean "demonstration of code correctness." In numerical simulation of physical models, verification is a non-trivial exercise, because the point of most simulation codes is to approximate a solution *that is not known*. Verification is itself subdivided into two activities that have different relationships to UQ.

Code Verification is the process of formally verifying the convergence of a discrete model to the continuous model being approximated. To accomplish code verification, one must have one or more *known* exact or approximate (with known approximation error estimates) solutions to the continuous model. Most frequently, a sequence of mesh refinements is used, and the behavior of the approximate solution error with mesh size is determined to see if it converges at the expected rate of convergence based on *a priori* error analysis of the numerical algorithm. Note that this implies, for rigorous code verification, that the continuous and discrete models are both known and well documented. Code verification should be done before any UQ analysis is done and is typically integrated into the software development process.

Calculation Verification, in contrast, is the estimation of the numerical approximation errors in the results of a simulation for which no exact solution exists. Thus, this type of verification is a quantification of the uncertainty in the results due to discretization. A key component of calculation verification is the application of techniques for *a posteriori* error estimation.

Benchmarking, that is, running two or more independent codes on the same problem and comparing the results, is a common confidence-building activity in scientific software, but it is not as rigorous or useful as formal verification techniques. Fundamentally, both codes may converge to the same answer, but it is not necessary that this answer is a good approximation to a solution of the governing equations.
When one of the codes involved in benchmarking has undergone extensive verification, this increases the utility of the benchmarking results. However, benchmarking cannot in general provide demonstrations of correct rates of convergence; and when discrepancies occur, it may be ambiguous as to which code needs improvement. Code verification is thus preferred over benchmarking.

For more detailed discussions of these ideas, many sources exist. Two references that we have found particularly useful in providing technical detail on methods and experience in UQ and SA are the two review articles authored by J. C. Helton and co-workers [1,2]. These two reviews contain over 500 references in total. The text by Roach [3] on Verification and Validation is also a venerable source on these basic concepts as well as techniques.

2.2 Process

Prior to conducting a UQ analysis of a simulation, a systematic code verification study should be done. The uncertainty quantification process is then broadly comprised of following steps:

1. Identification, definition, and documentation of quantities of interest from the computation that are measurables

2. Definition and documentation of the UQ study, which includes the identification of all parameters within a simulation code and reasonable bounds over which these parameters can be varied
3. Execution of the UQ study
4. Analysis and documentation of the results

It is often the case that steps 2-4 are iterated. The primary goal by the end of step four is to produce uncertainty estimates in the quantities of interest. An additional product will be sensitivity analysis results that can inform decisions on model and data improvement.

Code Verification

Code verification is generally conducted as part of the testing phase of software development. The goal is to demonstrate that the numerical method is coded as intended and that its solutions converge to solutions of the governing continuous model at the expected rate. Consider a linear model of numerical error suitable when errors are small:

$$\varepsilon_{\text{numerical}} = \varepsilon_{\text{discretization}} + \varepsilon_{\text{iteration}} + \varepsilon_{\text{roundoff}} + \varepsilon_{\text{programmer}} .$$

This model decomposes numerical error into error from discretization, error remaining from incomplete convergence of iterative algorithms, error from finite-precision arithmetic, and finally errors due to programming mistakes (bugs). Code verification relies on known analytically-derived solutions to compute the total numerical error (left-hand side), and because these solutions often are the result of model simplifications, it is seldom the case that code verification of physical simulation codes is exhaustive. The method of manufactured solutions [3] provides a less restrictive set of reference solutions, but application to coupled multi-physics codes requires substantial effort.

Most frequently, code verification is accomplished using convergence studies (Richardson Extrapolation) for each of the discretization parameters (mesh size, number of modes, order of interpolation, the time step, the statistical resolution or sample size, etc.), in the code. The sensitivity of results to changes in other numerical parameters, such as iteration tolerances, is also identified; and these parameter values are reduced so as not to mask the effects of discretization error. When completed, code verification identifies that the discretization error dominates the numerical error and that the discretization error converges at the a priori estimated rate. It is preferable that problems similar to those for the intended code use are used, although there may not be test problems suitably comprehensive for this. The governing equations, the numerical algorithm, the expected error behavior (i.e., a priori error estimates) and the code verification test problems, procedures, and results should be well documented. Often, completed code verification test cases are incorporated into (automated) tests suites for regression testing.

Determination of the Quantities of Interest

The first step in UQ analysis of a specific problem is to define and document one or more *quantities of interest (QOI)* or *metrics*. Simulations are run in order to compute specific quantities that are typically functionals of the solution. Quantifications of uncertainty only make sense in reference to these desired metrics. Examples of quantities of interest include point-wise values, such as the thermodynamic state at some location and at some time in a device; integrated quantities, such as the energy flux through a surface; and other derived quantities, such as the frequency of a driven response. Precise definitions of these metrics are required and must be documented.

Definition of the UQ Study

Once QOI are identified, an effective UQ study is defined. The inputs and parameters in the discrete model must be identified and valid ranges for these parameters are obtained. The common practice is to segregate the *numerical* and *physical* (or *model*) parameters. Numerical parameters are those associated with the discretization such as mesh resolution, time step, the order of the finite elements or interpolation, the statistical resolution if Monte Carlo methods are employed, the convergence tolerance of iterative methods, the effects numerical viscosity, etc. Physical parameters are frequently experimentally-derived inputs, such as transport coefficients, parameters in experimentally-fit constitutive relations, initial conditions, boundary conditions, etc. Physical parameters typically have some experimental uncertainty associated with them.

We note the distinction between numerical and physical model parameters. Numerical parameters (e.g., the number of cells, an iteration tolerance, the maximum CFL, etc.) possess no uncertainty but generate uncertainty in the discrete solution in the form of numerical approximation errors. Physical model parameters (e.g., transport coefficients, parameters for equations of state, initial conditions, etc.) have inherent uncertainty, and it is the sensitivity of the discrete solution to these physical parameters, combined with their uncertainties, that leads to their contribution in the overall uncertainty of the simulation result. In some sense, the numerical parameters lead to an "approximation" uncertainty while the physical model parameters have uncertainty that is "transferred" to uncertainty in the end result.

With parameters identified, the next step in study definition is to identify appropriate sensitivity analysis techniques. For numerical parameters, and in particular for numerical discretization error (i.e., calculation verification), the techniques commonly come from differential analysis methods. For physical parameters, however, statistical methods are often employed. In most cases, multiple simulations are required to generate the data for analysis. The number of simulations required scales nonlinearly with the number of parameters; the dimensionality of the parameter space grows geometrically with the number of parameters.

Multi-physics simulation codes have dozens to thousands of parameters. While some of the parameters may not be independent, this situation still leads to very high-dimensional parameter space. Interrogating (sampling) this parameter space can be prohibitively expensive; this problem is commonly referred to the *Curse of Dimensionality*.

To simply illustrate a principal and most obvious issue associated with the dimensionality of the UQ parameter space, if there are N parameters and each must take on M values to span a range of variation, then there are M^N possible unique combinations of parameters, i.e., the number of computations grows geometrically with the dimensionality of the parameter space. This quickly becomes an impractically large number of simulations for complex physical phenomena. Reduction of the computational burden becomes imperative.

If some of the parameters have very little influence or are constrained to vary insignificantly, then the dimensionality can be reduced. For some equations sets the parameter space can be much reduced by employing a careful scaling analysis as in the work of J. W. Connor and J. B. Taylor [4], who used a careful analytic treatment to determine the scaling of plasma fluid equations supporting turbulence and the minimum number of free parameters. Furthermore, experience with the model equations may indicate that the sensitivity of some of the parameters is quite non-uniform, which dictates that the sampling should be adjusted to be non-uniform. Sometimes these properties are not well understood until one is in the midst of the matrix of calculations, and one then adapts the sampling to take advantage of knowledge acquired from the evolving data set of computations.

For statistical approaches to sensitivity analysis, some random sampling of the space of inputs is required that covers the space of inputs in some fashion that does not corrupt or bias the determination of the relation of the outputs to the inputs. If a numerical approach is to be taken to determine the error or sensitivity analysis for a simulation of a complex physical system with more than a few physical and numerical parameters, then one needs to efficiently sample the parameter space of the control variables (physical and numerical). Some of the popular sampling choices are as follows: random sampling, stratified sampling, and Latin hypercube. Further examples of the sampling strategies are given in [1] and references therein.

When using uniform sampling, each parameter in the set of inputs is sampled to cover a specified range with a defined resolution. To explore all possible combinations of input parameters, the number of simulations grows rapidly with the dimensionality of the total parameter space.[1,2] An alternative to this is to consider the total parameter space as a whole and sample it randomly depending on the approximate uniformity of a random number generator used in the sampling, which uniformity improves as the inverse of the square root of the sample size. A variation of random sampling is stratified sampling [1] in which subdomains of the multi-dimensional parameter space are identified for a random sampling with higher average sampling densities than in other parts of the parameter space.

A more efficient sampling strategy than simple random sampling is the Latin hypercube sampling method which leads to variance reduction in the sampling [1] and allows fewer samples. The simplest illustration of the Latin hypercube sampling methodology is as follows. Consider a two-dimensional space of input parameters. If we sample each parameter with N_i sample values evenly distributed over the range of parameters, then there will be N_1N_2 possible combinations and calculations to undertake with uniform sampling. In a Latin hypercube sampling we might select N=(N1,N2)$_>$ total samples and select N values of the first parameter x_1 in a completely random order with respect to the index order of the values and the same number of N values of the second parameter x_2 using a completely random order that is uncorrelated with the order of the first parameter values. This set of values $\{x_1,x_2\}$ spans the two-dimensional parameter space in an optimal manner for the size N of the set of input pairs, and Latin hypercube sampling achieves a variance reduction as compared to random sampling [1].

Finally, prior to conducting a sensitivity analysis study, it is good practice to identify the necessary diagnostics and develop appropriate post-processing tools. The post-processing of simulation results often involves a large ensemble of parallel tasks, and the difficulties of large data set manipulation may necessitate direct integration of post-processing within the simulation. In addition, consideration of the experimental diagnostics may dictate the development and inclusion of synthetic diagnostics in the simulations. A proper accounting of all of the data necessary for analysis must be made in order to ensure that it is appropriately collected and archived.

Study Execution

As mentioned previously, the execution of a UQ study on a multi-physics code generally involves the execution of many simulations. This in turn generates tremendous amounts of data. It can be very difficult to complete such an undertaking manually without error. Thus, it is important to use tools (scripts, test harnesses, pipelines, etc.) that can manage the study execution efficiently and with as much automation as possible. Results must be appropriately documented and stored for future reference.

We note that studies cannot be conducted without human intervention for multi-physics codes. It often is the case, particularly with large, coupled code packages, that some simulations will not return results. Human intervention is required to determine if these results are due to human error, hardware error, violation of model applicability, coding errors, etc.

Analysis

Once the data is generated, the appropriate set of analysis tools can be applied to determine from the data, the sensitivities with respect to the numerical and physical parameters and ultimately the uncertainties associated with each of the uncertain inputs to the simulation. We discuss in more specificity such techniques in the next section. A simulation result with well-characterized uncertainties can then be compared with experimental data for validation.

2.3 Error and sensitivity analysis methodologies, statistical comparisons, regression and correlation analysis

Some of the many methodologies for analyzing error and parameter sensitivities are presented in [1,2]. In [1], the following methods are defined along with references: Monte Carlo analysis (statistically based methods for constructing probability distribution functions relating output/results to inputs/parameters), differential analysis (forward sensitivity analysis and adjoint methods [5,6]), response surface methodology (the output/results are related to the inputs/parameters via a reduced model that captures the multi-variable linear regression, for example), and Fourier amplitude sensitivity test and Sobol' variance decomposition (involving the decomposition of the variance in the output/results relative to the expectation of the output/results with respect to components due individual variables and interactions of the variables). The discussion of analysis techniques is extended in [2] to include scatter plots of the results with respect to the inputs, correlation analysis (construction of correlation coefficients between output and input variables), regression analysis (suitably normalized), partial correlation analysis (introduces some corrections in determining the relation of a specific output y_i to a specific input x_i after removing the linear affect on the output due to the other input parameters $x_{j \neq i}$), rank transformations (a means for transforming a nonlinear, but monotonic relation between outputs and inputs into a linear relation), statistical tests for patterns based on gridding, entropy tests for patterns based on gridding, nonparametric regression (an alternative to parametric regression procedures and grid-based procedures using an iterative approach to construct a model that captures the relations between the outputs and the inputs), squared rank differences/rank correlation coefficient, two-dimensional Kolmogorov-Smirnov (KS) test (an alternative to grid-based methods), tests for patterns based on distance measures (another alternative to grid-based methods), top-down coefficient of concordance, and complete variance decomposition. The discussion of these methodologies consumes most of [2].

In regards to techniques for verification, differential analysis tools are often employed; and in some sense, these all represent a form of *a posteriori* error estimation. The simplest, and most common, is Richardson extrapolation [3], which can be formulated in terms of mesh or order adaptation. In terms of the former, this is the basis of grid convergence studies and can be used for calculation verification provided three (or preferably more) resolutions are used to estimate the numerical error. Adjoint techniques for sensitivity analysis can also be formulated to determine the error in a specified quantity of interest [7]. These adjoint techniques can be applied either in terms of the discrete or continuous adjoint, although there are more theoretical gaps with the former formulation. Direct error evolution, also known as error transport methods, discretize and evolve equations for the error concurrently with the discrete solution [8,9]. Other techniques, such as residual and recovery methods [10], are particular to certain types of finite element discretizations. With the exception of Richardson extrapolation, all of these methods are active areas of research; all have been applied to non-trivial physical simulations, but the mathematical theory necessary to ensure reliability of these estimators in many cases is still under development

3. LLNL UQ EXPERIENCE AND EXAMPLES OF USE

There are many years of experience in uncertainty quantification in physical systems, e.g., experiments and experimental design. There is less experience in UQ in large-scale simulations, and UQ in large-scale simulations is a rapidly growing research area. UQ must become an essential element in producing science from simulations and in undertaking verification and validation tasks. Here we report some experience in UQ in simulation activities supporting large research programs at the Lawrence Livermore National Laboratory (LLNL), *e.g.*, in the areas of the National Ignition Campaign (NIC), weather modeling, and the Advanced Strategic Computing Program (ASC Program). At LLNL there is currently a large three-year Laboratory Directed Research and

Development Strategic Initiative project led by Richard Klein involving 20 scientists from four major departments of LLNL developing and applying UQ methodologies primarily to climate prediction. The developments in this LLNL Strategic Initiative will be carried into the area of Science Based Stockpile Stewardship, ICF capsule design and Nuclear Reactor design in the short term and progress into other fields involving large-scale simulation with multi-physics codes in the long term (e.g., astrophysics). Because UQ is an evolving research area, we can only give a snapshot in time of examples of use and comment on successful practices.

Some of the principal examples of UQ activity at LLNL are associated with calculations addressing the science of Stockpile Stewardship, NIC capsule implosion calculations [11,12], and climate modeling involving simulations with the Community Atmosphere Model and the Community Climate Modeling Suite. The codes used in these projects are large multi-physics codes, and massively parallel computing on high performance systems is employed. Radiation/hydrodynamics or Navier-Stokes equations or Monte Carlo computations in two and three spatial dimensions are involved. The calculations typically progress through several steps: code verification (against analytical calculations or an accepted benchmark computation arrived at independently), end-to-end sensitivity analysis and supporting computations using UQ methods, a matrix of computations using UQ methods to validate experiments, and use of UQ methods and results to assist with prediction of experimental behavior. In calculations exploring NIF capsule implosions at Livermore, Latin hypercube sampling is used, and data mining techniques are being used to analyze multivariate sensitivity data. Two-dimensional radiation/hydrodynamics multi-physics simulations are undertaken running on 64 cores per run with an ensemble of simultaneous runs (e.g., 100 – 200 runs) wrapped in a single MPI job using the UQ pipeline described in the next paragraph. A response function is synthesized ("trained") to approximate the application results over the input parameter domain. Experimental performance metrics are generated via nonlinear regression on a subset of the output features of the simulations. Statistical analysis to estimate probabilities and confidence in predictions of experimental behavior are based on use of the response function or performance metrics. In general, the researcher must understand the uncertainties in the experimental data quantitatively just as well as in the computations part of the validation process. The validation effort should involve the implementation and use of synthetic diagnostics in the computer simulations where appropriate.

At LLNL, a UQ computational "pipeline" is under development. The pipeline is driven by coding written in Python to provide a flexible user interface. The user can use the pipeline to launch and manage a suite of simulations exploring parameter space. The user selects a sampling strategy and supplies a basic input data deck, code run instructions, and the path to the application code. The pipeline undertakes submitting and managing the many jobs. The pipeline also contains a number of analysis tools for the user, e.g., statistical analysis tools, dimensional reduction methods; construction of a response function surrogate, multivariate adaptive regression splines (MARS) analysis, Bayesian MARS, a variety of sensitivity analysis tools, an interface to Sandia's DAKOTA software system [13], etc. Currently there is a UQ pipeline development effort to automate the sampling strategy in the pipeline so that it is self-adapting and self-guiding to steer the sampling to the more interesting regions in the multi-dimensional parameter space to greatly reduce the number of samples needed to examine the complex topology and the response functions of high-dimensional spaces.

The UQ pipeline is currently being used to explore uncertainties in the model predictions of the performance of the ignition capsules for NIC, the quantification of uncertainty in defense programs systems performance, and uncertainty in climate prediction. The UQ pipeline is being used to define the strategy for conducting the NIC experiments and to optimize the target design and laser pulse requirements for ignition. Under lessons learned at LLNL in using the UQ pipeline, it is important that the simulation application is robust over the input parameter domain; this can be the single most difficult task. Also, the data produced by the pipeline is voluminous, and retrieval of data from a storage archive tedious and slow. Thus, immediate comprehensive postprocessing of the ensemble of simulations in the pipeline is important.

The UQ pipeline is inheriting some functionality from the PSUADE software toolkit developed in the LLNL Computation Directorate for performing various uncertainty quantification (UQ) tasks. This toolkit includes functionality for forward uncertainty propagation, qualitative and quantitative sensitivity analysis, parameter exploration, risk analysis, and calibration/numerical optimization. PSUADE is targeted for multi-physics/ multi-component applications that are characterized by large number of correlated uncertain inputs/outputs, computationally demanding simulations, and different levels of available observational data. The toolkit has been applied to a variety of multi-physics applications including climate, subsurface flow, environmental management, structural analysis, and stockpile stewardship.

PSUADE employs the non-intrusive or sampling-based approach to UQ which does not require user codes to be modified. This "black-box" approach ensures that PSUADE can easily be integrated with application simulators. PSUADE offers a rich set of sampling designs for different types of analysis, a job execution environment for handling a large number of simulation runs, and many statistical analysis tools. One technology that is emphasized within PSUADE is the response surface methodology that can dramatically speed up the uncertainty quantification of large-scale models by taking advantage of the smoothness of the output function. As such, PSUADE is equipped with many response surface generation and validation techniques. These techniques can be coupled with other UQ techniques such as numerical optimization and Markov Chain Monte Carlo methods for calibration and parameter estimation.

In the area of code and calculation verification, the most common methodology in use at LLNL is Richardson extrapolation based on grid convergence studies. These are typically undertaken separately from physical parameter UQ. Because of limitations inherent in the assumptions of Richardson extrapolation, researchers in the Computation Directorate working in the UQ Science Strategic Initiative at LLNL are actively investigating adjoint and error transport methods for a posteriori error estimation of multi-physics applications.

4. SUMMARY AND RECOMMENDATIONS

To understand the results of the computations quantitatively and efficiently, and to quantify comparisons of simulations to experiment, the Fusion Simulation Program must adopt a careful UQ strategy. Based on our experience and assessment of UQ activities at LLNL, we make the following recommendations:

1. **The FSP should leverage existing UQ technologies.** This is not to say that UQ science and tools are fully developed; the science of UQ continues to evolve and there are still many open research issues [14-16]. Moreover, there is no document claiming to provide an authoritative summary of best practices in UQ. However, much relevant experience and many useful tools exist, such as described in Sec. 2 and 3 and in the UQ literature, which collectively should provide a starting point for FSP researchers.
2. **The FSP should integrate common UQ enabling technologies into the FSP framework.** This integration will facilitate UQ within the project by simplifying the execution and analysis of UQ studies and will lead to standardization across the project.
3. **The FSP should fund some level of UQ science research and development to augment existing techniques**. Gaps exist in the application of rigorous UQ techniques to complex, multi-physics codes, and these will be exacerbated by the component integration goals of the FSP. It is reasonable to assume that the FSP will face some unique challenges in the application of UQ that will be best addressed by a research effort on these issues within the FSP.
4. **The FSP should solicit input on UQ, or at the very least, comments on this document from other DOE Labs.** Both LANL and SNL, in particular, have well-established UQ efforts and could provide additional recommendations and perspectives.

In addition, we propose the following preliminary prescription for undertaking a vigorous UQ campaign for a given computational application. We assume that the code has undergone a thorough code verification process:

i. For the given application, identify and document all relevant quantities of interest. Define appropriate diagnostics for the results of the computations.

ii. Identify all the relevant physics and numerical parameters, and any other potential sources of error.

iii. Apply calculation verification methods to the numerical solution of the model equations. Adjust numerical parameters as necessary to ensure suitably accurate and converged results.

iv. Define the parameter space to be explored for the computational sensitivity and error analysis. Use analysis, e.g., scaling analysis, and *a priori* knowledge to reduce the dimensionality of the parameter space to be explored as much as possible.

v. Select an appropriate sensitivity analysis technique given the problem and available tools.

vi. Determine an efficient, rigorous, and practical evaluation (e.g., sampling) strategy for the numerical UQ campaign and define what data must be extracted from the simulation results for subsequent analysis. Define what data must be archived for future use in analysis.

vii. Execute the sensitivity analysis: employ a UQ pipeline or other enabling technology to automate the examination of parameter space, the quantitative analysis of the results, and the archiving of data.

viii. Use the raw data from the simulation results and the derived results obtained from analysis to undertake validation against experimental data or design and predict new experiments. The validation step likely will involve the exercise of many UQ analysis tools.

This process reflects the current state-of-the-art for studies conducted a LLNL; however, it is to be seen only as a recommendation for starting point. There are still improvements to be made in the process. For instance, as outlined, the process implicitly assumes that numerical errors can be minimized by a suitable choice of numerical parameters and thus eliminated from consideration in the subsequent quantification of uncertainty. Unfortunately, for coupled multi-physics codes, this is seldom the case; and numerical errors can potentially introduce sensitivity in the results comparable to the sensitivity with respect to the dominant physical parameters. Care must be taken to include numerical error estimates in such uncertainty quantifications, although precisely how to do this, particularly with the results of statistical sensitivity analyses, is still an open question.

This work was performed under the auspices of the US Department of Energy by the Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344 and was funded by DOE Fusion Energy Sciences as part of the Fusion Simulation Program. We are grateful to Scott Brandon and Brian Spears for assistance.

References

[1] J. C. Helton and F. J. Davis, “Latin Hypercube sampling and the propagation of uncertainty in analyses of complex systems,” Reliability Engineering and System Safety **81**, 23 (2003).

[2] J. C. Helton, J. D. Johnson, C. J. Sallaberry, and C. B. Storlie, “Survey of sampling-based methods for uncertainty and sensitivity analysis,” **91**, 1175 (2006).

[3] P. J. Roache, *Verification and Validation in Computational Science and Engineering,* Hermosa Publishers, Albuquerque, NM, 1998.

[4] J. W. Connor and J. B. Taylor, Nuclear Fusion, vol. 17, Oct. 1977, p. 1047-1055.

[5] M. B. Giles and N. A. Pierce, “An introduction to the adjoint approach to design,” Flow Turb. Comb. **65**, 393 (2000).

[6] S. Li and L. Petzold, “Adjoint sensitivity analysis for time-dependent partial differential equations with adaptive mesh refinement,” J. Comput. Phys. **198**(1), 310-325, (2004).

[7] D. Estep, M. Larson and R. Williams, “Estimating the error of numerical solutions of systems of nonlinear reaction-diffusion equations ,” *Memoirs of the American Mathematical Society* **696**, 1-109 (2000).

[8] I. Celik and G. Hu, “Single grid error estimation using error transport equation,” J. Fluids Eng. **126**, 778-790 (2004).

[9] A. Hay and M. Visonneau, “Error estimation using the error transport equation for finite-volume methods and arbitrary meshes,” Int. J. Comput. Fluid Dyn. **20**(7), 463-479 (2006).

[10] M. Ainsworth and J. T. Oden, *A Posteriori Error Estimation in Finite Element Analysis*, John Wiley & Sons, Inc., New York, 2000.[11] S. W. Haan, M. C. Herrmann, J. D. Salmonson, P.A. Amendt, D. A. Callahan, T. R. Dittrich, M. J. Edwards, O. S. Jones, M. M. Marinak, D. H. Munro, S. M. Pollaine, B. K. Spears, and L. J. Suter, “Rev 3 Update of Requirements for NIF Ignition Targets”, Fusion Sci and Tech. **55**, 227 (2009); “Update on design simulations for NIF ignition targets, and the rollup of all specifications into an error budget,” Eur. Phys. J. D **44** 249 (2007); “Update on specifications for NIF ignition targets,” Fusion Sci. and Tech. **51**, 509 (2007); and “Update on specifications for NIF ignition targets,” Fusion Sci. and Tech., 49, 553-557, (2006)

[12] Landen OL, Boehly TR, Bradley DK, Braun DG, Callahan DA, Celliers PM, Collins GW, Dewald EL, Divol L, Glenzer SH, Hamza A, Hicks DG, Hoffman N, Izumi N, Jones OS, Kirkwood RK, Kyrala GA, Michel P, Milovich J , Munro DH , Nikroo A, Olson RE, Robey, Spears BK, Thomas CA Weber SV, Wilson DC, Marinak MM, Suter LJ, Hammel BA, Meyerhofer DD, Atherton J , Edwards J, Haan SW, Lindl JD, MacGowan BJ, Moses E, "Capsule performance optimization in the National Ignition Campaign", Phys Plasmas **17,** 056301 (2010).

[13] http://dakota.sandia.gov/software.html

[14] http://www.begellhouse.com/journals/52034eb04b657aea

[15] D. Higdon and R. Klein, "Uncertainty Quantification and Error Analysis," in *Scientific Grand Challenges for National Security: The Role of Computing at Extreme Scale*, Report of the Workshop Held October 6-8, 2009, DOE, Washington, D.C.

[16] R. Klein, "Uncertainty Quantification," in *Report on the Workshop on Accelerated Nuclear Energy Materials Development*, June 2010, Lawrence Livermore National Laboratory Technical Report (LLNL-TR-436353).

UQ Contacts at LLNL

Scott Brandon
David Domyancic
Jeffrey Hittinger
Gardar Johannesson
Richard Klein
Brian Spears
Charles Tong
Carol Woodward

Other Literature Acquired

1. K. Walter, "Narrowing Uncertainties," Science & Technology Review, July/August 2010, p. 12, Lawrence Livermore National Laboratory.

2. B. H. Thacker, S. W. Doebling, F. M. Hemez, M. C. Anderson, J. E. Pepin, E. A. Rogriguez, “Concepts of Model Verification and Validation,” Los Alamos National Laboratory Report (October 2004), LA-14167-MS.
3. M. Pilch, T. Turcano, J. Moya, G. Froehlich, A. Hodges, and D. Percy, “Guidelines for Sandia ASCI Verification and Validation Plans – Content and Format: Version 2.0,” Sandia National Laboratories Report (January 2001), SAND2000-3101.
4. P. Strand, C. Angioni, C. Bourdelle, J. Lister, F. Rimini, M. Romanelli, and H. Weisen, “DRAFT Guidelines for the Validation and Verification procedures,” European Fusion Development Agreement Report, EU-ITM-TF (04)-08.
5. J. G. Yost, et al., “Guide for Verification and Validation in Computational Solid Mechanics,” The American Society of Mechanical Engineers, December 2006, ASME V&V 10-2006.
6. R. Klein, S. Doebling, F. Graziani, M. Pilch, T. Trucano, “ASC Predictive Science Academic Alliance Program Verification and Validation Whitepaper,” Lawrence Livermore National Laboratory Technical Report (UCRL-TR-220342-Rev), March 29, 2006.
7. M. Stein, “Large Sample Properties of Simulations Using Latin Hypercube Sampling,” Technometrics, **29**, 143 (1987).
8. L. J. Lucas, H. Owhadi, and M. Ortiz, “Rigorous verification, validation, uncertainty quantification and certification through concentration-of-measure inequalities,” Comput. Methods Appl. Mech. Engrg. **197**, 4591 (2008).
9. C. S. Woodward, K. E. Grant, R. Maxwell, “Applications of Sensitivity Analysis to Uncertainty Quantification in Variably Saturated Flow,” in Proceedings of the XIVth International Conference on Computational Methods in Water Resources (CMWR XIV), Vol **47**, 73 (2002).
10. R. W. Logan, C. K. Nitta, S. K. Chidester, “Estimating Parametric, Model Form, and Solution Contributions Using Integral Validation Uncertainty Quantification,” Lawrence Livermore National Laboratory, Technical Report (March 17, 2006), UCRL-TR-219879, in 47th AIAA/ASME/ASCE/AHS/ASC Structures, Structural

Dynamics, and Materials Conference, 1 - 4 May 2006, Newport, Rhode Island, AIAA 2006-1923